Solaris Performance Diagnostics for Database People

Dallas Deeds
Nationwide

Speaker Background

- Dallas Deeds, DBA at Nationwide for 11 years
- Primary focused on performance for last 9 years
- 15 years IT experience

Nationwide®
On Your Side

The Shop

7 Production DBAs

815 Oracle databases (147 RAC)
- sizes ranging from 1 GB to 8 TB

Primarily on Solaris machines
- A sprinkling of HP, AIX, Linux

Primarily frame storage

What we're covering

- Why look at servers?
- What to look at
- Tools to look at things
- How to use those tools
- Example output
- What some of the metrics mean

Tiers

- Database
- Database server
- Application
- Clients
- App Server(s)
- Web Server(s)
- These include all the connectivity between tiers
- Can be quite time consuming

Why look at servers?

- What do (Oracle) databases do?
  - get requests from users
  - parse SQL
  - run code (PL/SQL, Java, etc)
  - look to see if things are in memory
  - read blocks from disk
  - write those blocks into memory
  - write data to disk
  - interact with local or remote databases
  - send data to users
- Some of the above is CPU
- The rest is disk, waits, messaging or server/client traffic

Why look at servers, cont'd

- Your databases rely on hardware
  - You need to know what's going on
  - You need to be able to speak the lingo
  - You don't want to learn it when you are already having problems

Basic assumptions

- You will likely look at this while you look at your usual things
  - your favorite v$ views
  - 10046 trace data
  - Runes, bones, tea leaves
- Especially when there isn't an obvious fix
- Say you are doing tons of db file s* reads
  - Are they slow? How do you know?
  - Are your devices overloaded?

What to look at?

- What do you look at outside the database?
- On the database server
  - Resources that the database needs
    - CPU
    - Memory – physical and virtual
    - I/O subsystems
    - Network
  - Don't forget the other machines your app needs

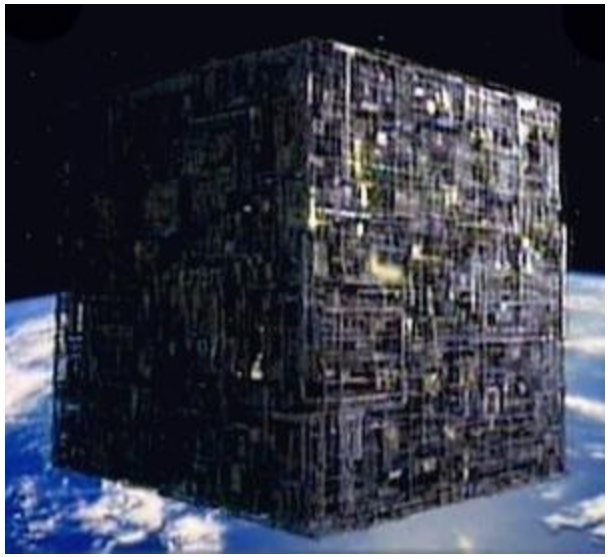
What to look at (cont'd)

- Find out why you need so much of a particular resource
- There's probably a fair amount of waste
- There's probably something silly that snuck past (nearly) everyone

Black Boxes

- We've been taught not to treat the database as a black box
- Why treat servers that way?

!=



So what's in the black box

- /usr/platform/sun4u/sbin/prtdiag
- The "sun4u" piece can change
  - Some new servers may not be sun4u
    - Ultrasparc T1 & T2 systems are "sun4v"
- /usr/platform/`uname –m`/sbin/prtdiag

Prtdiag on single-core

```
> /usr/platform/`uname -m`/sbin/prtdiag | more
System Configuration:  Sun Microsystems  sun4u Sun Fire 880
System clock frequency: 150 MHz
Memory size: 8192 Megabytes

========================= CPUs ============================

             Run   E$  CPU    CPU
Brd  CPU    MHz   MB Impl.   Mask
--- ----- ---- ---- ------- ----
 A    0   1200  8.0 US-III+ 11.1
 B    1   1200  8.0 US-III+ 11.1
 A    2   1200  8.0 US-III+ 11.1
 B    3   1200  8.0 US-III+ 11.1
```

We have 4 CPUs,8 GB of memory on this machine

Prtdiag, dual-core machine

```
> /usr/platform/`uname -m`/sbin/prtdiag | more
System Configuration:  Sun Microsystems  sun4u Sun Fire V890
System clock frequency: 150 MHz
Memory size: 65536 Megabytes

========================= CPUs ============================

             Run   E$  CPU    CPU
Brd  CPU    MHz   MB Impl.   Mask
--- ----- ---- ---- ------- ----
 A   0, 16 1800 32.0 US-IV+   2.2
 B   1, 17 1800 32.0 US-IV+   2.2
 A   2, 18 1800 32.0 US-IV+   2.2
 B   3, 19 1800 32.0 US-IV+   2.2
```

This server has 4 dual-core CPUs,64 GB of memory
Does not show mutli-threading capabilities – vmt nor cmt

Prtdiag - memory

```
========================= Memory Configuration ==========================
             Logical  Logical  Logical
      MC     Bank     Bank     Bank          DIMM    Interleave  Interleaved
 Brd  ID     num      size     Status        Size    Factor      with
---- ---    ----     ------   -----------   ------  ----------  -----------
  A    0       0       512MB   no_status      256MB     8-way         0
  A    0       1       512MB   no_status      256MB     8-way         0
  A    0       2       512MB   no_status      256MB     8-way         0
  A    0       3       512MB   no_status      256MB     8-way         0
  B    1       0       512MB   no_status      256MB     8-way         1
  B    1       1       512MB   no_status      256MB     8-way         1
  B    1       2       512MB   no_status      256MB     8-way         1
  B    1       3       512MB   no_status      256MB     8-way         1
  A    2       0       512MB   no_status      256MB     8-way         0
  A    2       1       512MB   no_status      256MB     8-way         0
  A    2       2       512MB   no_status      256MB     8-way         0
  A    2       3       512MB   no_status      256MB     8-way         0
  B    3       0       512MB   no_status      256MB     8-way         1
  B    3       1       512MB   no_status      256MB     8-way         1
  B    3       2       512MB   no_status      256MB     8-way         1
  B    3       3       512MB   no_status      256MB     8-way         1
```

Prtdiag - I/O cards

```
========================= IO Cards =========================

                                Bus   Max
      IO   Port Bus             Freq  Bus   Dev,
Brd   Type   ID  Side  Slot    MHz   Freq  Func  State Name                              Model
---- ---- ---- ----  ----    ----  ----  ----  ----- --------------------------------  ---------------------
I/O   PCI    8    B     3       33    33   2,0   ok    network-pci108e,abba.20           SUNW,pci-ce
I/O   PCI    8    B     2       33    33   3,0   ok    pci-pci8086,b154.0/scsi (scsi)    PCI-BRIDGE
I/O   PCI    8    B     2       33    33   4,0   ok    scsi-pci1077,1016/sd (block)      QLGC,ISP10160/pci-brid+
I/O   PCI    8    B     2       33    33   5,0   ok    scsi-pci1077,1016/sd (block)      QLGC,ISP10160/pci-brid+
I/O   PCI    8    B     1       33    33   4,0   ok    pci-pci8086,b154.0/pci108e,1000   PCI-BRIDGE
I/O   PCI    8    B     1       33    33   0,0   ok    pci108e,1000-pci108e,1000.1       device on pci-bridge
I/O   PCI    8    B     1       33    33   0,1   ok    SUNW,qfe-pci108e,1001             SUNW,pci-qfe/pci-bridg+
I/O   PCI    9    A     8       66    66   1,0   ok    lpfc-pci10df,fa00/sd (block)      LP10000
I/O   PCI    9    A     7       66    66   2,0   ok    lpfc-pci10df,fa00/sd (block)      LP10000
```

We have some qlogic cards and LP 10000s

Tools

- There are various industrial-strength tools
- You may own some already
  - BMC Best/1 (Perform & Predict)
  - Teamquest
  - Hyperformix
  - SiteScope
  - HP OpenView
- But be careful

Server diagnostics - some tools

- sar
- vmstat
- iostat
- mpstat
- top
- prstat
- netstat
- nicstat
- uptime
- truss
- DTrace

CPU

- Two measures
  - Utilization
  - Run queue
- Utilization is composed of
  - User
  - System
  - Idle
  - Wait for I/O*

CPU, (2)

- Run queue
  - Reflects CPU saturation
  - Processes on CPU plus waiting to get on CPU
  - 2 – 3 times # of CPUs (cores) is bad news
    - One reason why it is important to know how many CPUs you have in a server
    - Divide by # of CPUs to compare to other servers

CPU triage for fun & profit

- sar
  - shows utilization
- vmstat
- top
  - Available from unixtop.org
- prstat
  - Shows top processes, run queue (not utilization, though)
  - Various switches change what you see
- uptime
  - Shows run queue

sar

```
> sar -u 5 5

SunOS nessus 5.8 Generic_117350-39 sun4u    05/05/07

23:18:47    %usr    %sys    %wio   %idle
23:18:52      10       2      12      76
23:18:57      10       2      21      67
23:19:02      10       5       8      77
23:19:07      10       3       3      84
23:19:12      10       6       2      82

Average       10       4       9      77
```

Lots of headroom here – what does that get you?

CPU Headroom

- Allows you accommodate peaks..
  - Month-, quarter-, year-end
  - When someone adds 1,000 users and forgets to tell you
  - When SOX police require you to turn on an audit that is hideously expensive
- Too much headroom, bean counters think that they aren't getting their money's worth
- Too little, and half of your automated monitoring tools will page you to death.

sar, on a busy machine

```
> sar 1 5

SunOS pepperjack 5.8 Generic_117350-27 sun4u    02/11/08

15:26:38     %usr     %sys     %wio    %idle
15:26:39       71       28        1        0
15:26:40       77       23        0        0
15:26:41       77       23        0        0
15:26:42       81       19        0        0
15:26:43       75       25        0        0

Average        76       24        0        0
```

top – a busy system

```
load averages: 16.10, 15.76, 15.57                  newton                  12:52:21
266 processes: 243 sleeping, 12 running, 2 zombie, 1 stopped, 8 on cpu
CPU states:  0.0% idle, 82.6% user, 17.3% kernel,  0.1% iowait,  0.0% swap
Memory: 16.0G real, 11.3G free, 1.7G swap in use, 12.9G swap free

   PID USERNAME THR PR NCE   SIZE   RES STATE    TIME FLTS    CPU COMMAND
   139 oracle    11 31   0  384M  347M run    26:36   13  4.46% oracle
   322 oracle    11 31   0  384M  347M run    25:42   24  4.40% oracle
 29775 oracle    11 31   0  386M  349M run    25:41   26  4.33% oracle
 29438 oracle    11 31   0  386M  349M run    25:39   38  4.30% oracle
 29838 oracle    11 20   0  386M  349M cpu00  25:47   19  4.29% oracle
   315 oracle    11 31   0  385M  350M run    25:54   18  4.25% oracle
 29546 oracle    11 31   0  386M  349M cpu03  25:16   32  4.24% oracle
   197 oracle    11 31   0  387M  351M cpu16  24:54   14  4.21% oracle
 29717 oracle    11 31   0  387M  349M cpu01  25:18   42  4.12% oracle
   275 oracle    11 31   0  387M  350M run    25:31   35  4.11% oracle
 29922 oracle    11 31   0  386M  349M run    25:03   41  4.08% oracle
 29381 oracle    11 60   0  385M  352M sleep  25:50   25  4.07% oracle
 29492 oracle    11 41   0  386M  349M sleep  25:49   33  4.05% oracle
 29987 oracle    11 32   0  386M  349M cpu16  25:30   55  4.05% oracle
    36 oracle    11 31   0  387M  351M run    25:03   27  4.03% oracle
```

top, cont'd

```
top -c

load averages:  5.67,  6.29,  6.41;                 up 71+13:41:53            16:06:21
1378 processes: 1374 sleeping, 4 on cpu
CPU states:  5.3% idle, 69.7% user, 14.1% kernel, 10.9% iowait,  0.0% swap
Memory: 16G phys mem, 3734M free mem, 7172M total swap, 7172M free swap

   PID USERNAME LWP PRI NICE  SIZE   RES STATE    TIME    CPU COMMAND
  3491 pcmadm     5   0    0  312M  300M cpu/3  102.7H  8.77% scenter scheduler problem
 24006 pcmadm     4  30    0  314M  304M sleep    2:33  8.61% scenter -listener -xsocket:17,5
 24769 pcmadm     4  11    0  344M  335M cpu/4   32:39  6.89% scenter -listener -xsocket:17,5
  3689 pcmadm     5   0    1  310M  298M sleep  189.4H  6.79% scenter scheduler event.bridge -
    b
 11757 best1      1   0    0   14M 9904K sleep   88.3H  4.84% bgscollect -I noInstance -B
 22044 pcmadm     4  30    0  314M  304M sleep    2:04  3.83% scenter -listener -xsocket:17,5
  4022 oracle    14  60    0  926M  891M sleep    0:49  3.77% oraclescprod (LOCAL=NO)
  5696 oracle     1  59    0  926M  891M sleep    2:44  3.51% oraclescprod (LOCAL=NO)
   355 pcmadm     4  28    0  312M  302M sleep    0:08  3.38% scenter -listener -xsocket:17,5
 28191 oracle     1   1    0  925M  884M sleep  583:52  2.63% oraclescprod (LOCAL=NO)
 24771 oracle    11  60    0  929M  895M sleep   17:29  2.57% oraclescprod (LOCAL=NO)
 24008 oracle     1  20    0  926M  891M sleep    2:58  2.47% oraclescprod (LOCAL=NO)
```

Use two

- terminal windows, that is
- Run top in one
- Run SQL*Plus in the other
- Pick a top CPU hogging process
  - Map it to v$sql through v$process and v$session

Using two….

```
load averages: 15.66, 15.73, 15.61                    newton
   12:59:09
265 processes: 248 sleeping, 9 running, 1 zombie, 1 stopped, 6 on cpu
CPU states:     % idle,     % user,     % kernel,     % iowait,     % swap
Memory: 16.0G real, 11.8G free, 1.3G swap in use, 13.5G swap free

   PID USERNAME THR PR NCE   SIZE   RES STATE   TIME FLTS     CPU COMMAND
   139 oracle    11 10   0   384M  347M run    28:57    0   4.53% oracle
➔ 315 oracle    11 21   0   385M  350M run    28:15    0   4.51% oracle
 29600 oracle    11 21   0   385M  349M run    28:34    0   4.50% oracle
 29438 oracle    11 21   0   386M  349M run    27:59    0   4.41% oracle
 29838 oracle    11 21   0   386M  349M run    28:07    0   4.35% oracle
 29656 oracle    11 21   0   385M  347M sleep  27:44    0   4.33% oracle
 29381 oracle    11 11   0   385M  352M cpu18  28:10    0   4.29% oracle
   251 oracle    11 60   0   387M  349M sleep  27:54    0   4.24% oracle
   275 oracle    11 21   0   387M  350M cpu17  27:51    0   4.24% oracle
 29546 oracle    11 60   0   386M  349M sleep  27:32    0   4.22% oracle
 29922 oracle    11 21   0   386M  349M cpu19  27:20    0   4.21% oracle
 29717 oracle    11 60   0   387M  349M sleep  27:36    0   4.20% oracle
 29775 oracle    11 21   0   386M  349M sleep  27:59    0   4.18% oracle
 29492 oracle    11 12   0   386M  349M run    28:10    0   4.17% oracle
   322 oracle    11 12   0   384M  347M cpu02  28:00    0   4.14% oracle
```

Using 2… (cont'd)

```
select sql_text,executions,disk_reads,buffer_gets,address,hash_value
from v$sql
where address in (select sql_address
                    from v$session
                    where paddr = (select addr from v$process where spid = &1)
                  );

SQL> /
Enter value for 1: 315  ←
old   6: where spid = &1))
new   6: where spid = 315))

SQL_TEXT
--------------------------------------------------------------------------------
EXECUTIONS DISK_READS BUFFER_GETS ADDRESS          HASH_VALUE
---------- ---------- ----------- ---------------- ----------
select rowidtochar(rowid) ,varuvdef_seq_id ,fee_seq_id ,start_date ,end_date ,bu
siness_area ,msmtperd_type ,period ,frequency  from perf_calculation_queue where
 ((queue_id=:b1 and completion_datetime is null ) and calc_code is null ) order
by calc_priority,queue_datetime
      5153   25533219    33269567 00000003864AA258  688574372
```

prstat

```
   PID USERNAME  SIZE   RSS STATE  PRI NICE      TIME  CPU PROCESS/NLWP
   934 root      862M  702M sleep   59    0  57:18:36 8.7% vxsvc/53
 22263 oracle    783M  778M cpu3    43    0   5:14:16 7.2% oracle/30
 25106 oracle   6085M 6078M sleep   59    0   4:45:30 0.9% oracle/1
  9173 oracle   6085M 6077M sleep   59    0   8:59:30 0.6% oracle/1
 12725 root       20M   15M cpu0    18    0   0:00:00 0.5% symdisk/1
    56 root       12M 5440K sleep   49    0   0:39:53 0.3% vxconfigd/1
  2293 root     5200K 2848K sleep   59    0   1:46:53 0.2% MountAgent/13
 12663 oracle   6480K 6008K cpu2    59    0   0:00:00 0.2% prstat/1
  9175 oracle    773M  765M sleep   59    0   1:41:51 0.1% oracle/1

Total: 467 processes, 2778 lwps, load averages: 2.39, 2.07, 1.90
```

Why is vxsvc chewing up so much CPU?

Prstat, cont'd

```
> prstat -mL
   PID USERNAME  USR  SYS  TRP  TFL  DFL  LCK  SLP  LAT  VCX  ICX  SCL  SIG PROCESS/LWPID
 28122 oracle     35   26 0.0  0.0  0.0  0.0   44  0.0  174   1K  20K    0 oracle/1
 19525 oracle     47  6.2  0.2  0.0  0.0  0.0  4.5   42   51   1K  827    0 oracle/1
 28090 oracle     38   13 0.0  0.0  0.0  0.0   49  0.0  190   1K   5K    0 oracle/1
 23202 oracle     44  5.7  0.2  0.0  0.0  0.0  4.8   46   29   2K  635    0 oracle/1
 25176 oracle     38  9.6  0.2  0.0  0.0  0.0  6.0   46   38   1K  19K    0 oracle/1
 28080 oracle     41  6.1  0.1  0.0  0.0  0.0  4.1   48   64   1K  31K    0 oracle/1
 28020 oracle     38  7.1  0.1  0.0  0.0  0.0   12   43  366   1K  12K    0 oracle/1
 28004 oracle     37  7.5  0.2  0.0  0.0  0.0  9.4   46  418   1K   1K    0 oracle/1
 27671 oracle     42  1.7  0.2  0.0  0.0  0.0   26   30  256   1K   4K    0 oracle/1
 27567 oracle     33  8.8  0.1  0.0  0.0  0.0  6.4   51   68   1K  17K    0 oracle/1
 23390 oracle     36  6.0  0.1  0.0  0.0  0.0   11   47  104   1K  19K    0 oracle/1
 27048 oracle     35  4.4  0.1  0.0  0.0  0.0   25   36  196   1K  32K    0 oracle/1
 27861 oracle     33  6.9  0.2  0.0  0.0  0.0   18   42  181   1K  13K    0 oracle/1
 28086 oracle     18  4.8  0.0  0.0  0.0  0.0   78  0.0  329  375   3K    0 oracle/1
 27739 oracle     19  2.7  0.1  0.0  0.0  0.0   71  7.2   1K  858  12K    0 oracle/1
 28024 oracle    4.8  4.4  0.0  0.0  0.0  0.0   88  2.3  765  242   1K    0 oracle/1
Total: 165 processes, 1507 lwps, load averages: 14.32, 14.74, 14.71
```

A word of caution

- Know what overheads you are introducing
- Top can be intrusive
- Top and prstat use /proc
  - Prstat opens the filehandles once and reads
  - Top does an open, read and close for each process, on each refresh
    - This leads to a lot of extra syscalls
    - The overhead increases as you have more processes
- http://www.brendangregg.com/DTrace/prstatvstop.html

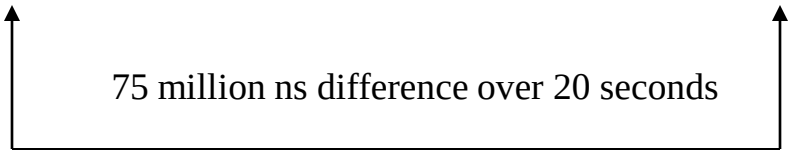
Procsystime comparison

top

SYSCALL	TIME (ns)
sysconfig	6400
gtime	7900
lseek	17200
uadmin	42500
pollsys	266500
write	543700
ioctl	674600
getdents	19581200
close	27900900
open	56760200
read	327501200
TOTAL:	433302300

prstat

SYSCALL	TIME (ns)
getloadavg	9400
lseek	26500
getpid	38800
fcntl	154300
close	453400
fstat	616700
write	762200
getmsg	854100
read	1738300
pollsys	2407800
putmsg	2487300
open	3802100
getdents	18843100
pread	325690400
TOTAL:	357884400



Memory

- Critical to Oracle servers
- 2 – 4 GB per CPU, ymmv
- You want some headroom
- You don't want to be scanning
- You don't want to run out of swap
- Memory can be controlled (to a point)

Paging and scanning

- Filesystem paging is normal
- Anonymous paging is not
  - Seen when physical memory is nearly or completely exhausted
  - The page scanner becomes active
    - Moves pages that haven't been used recently to swap
  - Next access requires processes (for which pages that have been swapped) to sleep while the pages are retrieved from swap and loaded into memory again
    - This is *not* particularly fast

vmstat

The first line is an summary since system startup, so you usually aren't interested in it

```
chinook:/u01/home/oracle> vmstat 5 5
 kthr      memory            page            disk          faults      cpu
 r b w   swap      free  re  mf   pi po fr de sr m0 m1 m2 m3   in   sy   cs us sy id
 1 0 0 16187312 15266736 745 1290 1958 396 394 0 0 4 4 0 0 9017 14560 6071 50 16 34
 0 0 0 15473896 14815736 487 1524 0 5 5 0 0 0  0  0  0 2351 6074 2978 30 21 49
 2 0 0 15737728 14802160 876 6434 2 0 0 0 0 0  0  0  0 2619 27248 2942 52 27 21
 0 0 0 15489176 14803744 543 2141 3 3 3 0 0 23 23 0  0 2708 8974 3131 36 23 41
```

top

```
load averages:  4.16,  3.84,  3.66                          bmw2           09:30:59
89 processes:  83 sleeping, 1 running, 1 stopped, 4 on cpu
CPU states:  1.1% idle, 76.1% user,  8.5% kernel, 14.2% iowait,  0.0% swap
Memory: 8.0G real, 115M free, 7.3G swap in use, 847M swap free
   PID USERNAME THR PR NCE  SIZE   RES STATE   TIME FLTS     CPU COMMAND
 17734 oracle    11  0   0  2.3G  2.3G cpu02 143:56   10 23.59% oracle
 17742 oracle    11  0   0  1.3G  884M cpu00 122:06   62 21.76% oracle
 17731 oracle    11  1   0  2.3G  2.2G cpu03 127:15  906 16.05% oracle
 17738 oracle    11  1   0  1.3G  927M run   131:56 1170 14.71% oracle
  3167 root      12 58   0 24.5M 9432K sleep 583:07    0  2.14% msragent
    24 root       1 48   0 10.7M 2288K sleep 451:26    0  0.25% vxconfigd
 26381 oracle     1 58   0 3168K 2240K cpu01   0:00    0  0.15% top
  4433 oracle     1 58   0  2.2G  174M sleep  10:20    4  0.07% oracle
  1422 root      17 59   0 4880K 3152K sleep 105:30    2  0.04% MountAgent
 29795 oracle     1 58   0  1.3G 34.5M sleep   2:30    0  0.02% oracle
   804 root       5 58   0 10.5M 2920K sleep  36:39    0  0.02% automountd
   838 root      12 52   0 3856K 2256K sleep  12:08    0  0.01% nscd
   630 root      15 41   0 8960K 2296K sleep 117:25    0  0.01% picld
   684 root      22 60   0  147M 11.7M sleep  14:28    4  0.01% vxsvc
  1424 root      17 59   0 4504K 2832K sleep   1:49    0  0.01% MultiNICAAgent
```

Sum the memory

- Two instances
  - Total SGA – 3.6 GB
    - SGA #1 – 1.3 GB
    - SGA #2 – 2.3 GB
  - Total possible PGA – 5.4 GB
    - PGA #1 – 2.0 GB
    - PGA #2 – 3.4 GB
- Total possible memory consumption
  - 9.0 GB
- Who remembers the total physical memory from the previous slide?
  - 8.0 GB
  - Whoops!
  - Oracle instances *alone* could potentially consume more memory than physically existed on the server

Going from busy to

```
vmstat -p
memory                  page                          executable      anonymous      filesystem
   swap  free    re  mf       fr     de  sr   epi  epo  epf  api  apo  apf  fpi  fpo  fpf
 1187936 128032 31  355       0      0   0     0    0    0    0    0    0    0    0    0
 1187888 127944 45  391      29      0   0     0    0    0    0    0    0    0   29   29
 1187976 127936 26  298       0      0   0     0    0    0    0    0    0    0    0    0
 1187864 127912 29  291       0      0   0     0    0    0    0    0    0    0    0    0
 1188096 128016 14  64       13      0   0     0    0    0    0    0    0    0   13   13
 1188088 128152 13  118      69      0   0     0    0    0    0    0    0    0   69   69
 1188080 128208 0   0        18      0   0     0    0    0    0    0    0    0   18   18
 1288928 228920 57  374      50      0   0    13    0    0    0    0    0   16   50   50
 1339352 279160 0   0         0      0   0     0    0    0    0    0    0    0    0    0
 1339352 279160 0   4         0      0   0     0    0    0    0    0    0    0    0    0
 1086376 47904 116 10749  4077      0 5082     0    0    2  216 3029 3000    0    0 1074
 1086264 39328 831   898  7866      0 5697     0    0   24  458 7338 7085   37   21  757
 1086264 47104 694   718 10370      0 5418     0    0   32  165 8448 8344    0    0 1994
 1086264 50704 1047 1056  8133      0 4988     0    0   29   16 7637 7485    0    2  618
 1086280 57832 1227 3037 10040  40864 6840     0    0 1154    8 9088 8282   13   24  602
 1086216 67400  877 2181 10229  52680 7902    34    0 2714    2 7040 6168    0    0 1346
 1085976 80288  471 1299  9994  74888 7786     0    0 1933    0 7040 6562    0   32 1498
 1085904 91792  993 1592  9133  91048 8968    24    0 2557   34 6314 5893    5   42  682
 1085848 111960 581 1087 12984  82704 9812     0    0 3856    8 8149 8112    0    2 1016
```

I/O

- This is where your database spends a lot of time
- A lot has been said about staying away from disk
  - Somewhat erroneously
  - Disk reads (PIOs) are always motivated by LIOs (Oracle buffer operations) – ditch one, ditch both
- It's all about workload
  - How much are you doing
  - How often do you do it

I/O, cont'd

- Application did 35 million PIOs, from 36 million LIOs. Per day.
- Why?
  - They decided to insert the output of a join of dba_users and v$sqlarea into a staging table
  - AFTER checking that the SQL address didn't exist in the staging table
- Guess what?
  - No indexes on SQL address, staging table was "from beginning of time"

I/O, cont'd

- Know your thresholds
- Currently, high-end frame-based storage
  - Can do well above 50 MB/second
  - Better be getting average service times < 30 ms
  - You need to be concerned with I/Os per second
  - And throughput
- Haven't had much experience with Tier 2 disk
  - Except when it goes horribly wrong
  - Can perform quite well if configured properly
  - Can be a nightmare otherwise

iostat

- First line is summary since boot
- Use small polling interval
- Some of the useful columns

```
r/s    number of reads per second
w/s    number of writes per second
kr/s   number of kilobytes read per second
kw/s   number of kilobytes written per second
wait   number of transactions waiting for service (wait queue)
wsvc_t average service time in wait queue, in milliseconds
asvc_t average service time of active transactions, in milliseconds
%w     percentage of time there are transactions waiting for service
%b     percentage of time the device is active for the observed
                interval; utilization for local storage
```

iostat, cont'd

- Storage arrays
  - `%b` not relevant for storage arrays
  - Throughput is the valid measurement
  - Throughput = `kr/s + kw/s`
- `Asvc_t`
  - Smaller numbers (faster) are better
- `Wsvc_t`
  - queueing

Iostat – useful switches

- -x extended device statistics
- -C report controllers and segregate by controller
- -n display descriptive names; like cXtYdZsN
- -z don't print all-zero lines
- -E show device error statistics

iostat example output

```
iostat -xnCz 1 | grep c
                    extended device statistics
    r/s    w/s   kr/s   kw/s wait actv wsvc_t asvc_t  %w  %b device
    0.0    1.0    0.0    8.0  0.0  0.0    0.0    5.5   0   1 c0
    0.0    1.0    0.0    8.0  0.0  0.0    0.0    5.5   0   1 c0t8d0
    0.0    1.0    0.0    8.0  0.0  0.0    0.0    2.5   0   0 c2
    0.0    1.0    0.0    8.0  0.0  0.0    0.0    2.5   0   0 c2t0d0
    0.0    5.0    0.0   48.0  0.0  0.3    0.0   50.3   0  24 c4
    0.0    1.0    0.0    8.0  0.0  0.1    0.0   83.7   0   8 c4t50060482D52D2596d6
    0.0    1.0    0.0    8.0  0.0  0.0    0.0   21.8   0   2 c4t50060482D52D2596d3
    0.0    2.0    0.0   16.0  0.0  0.1    0.0   49.3   0   9 c4t50060482D52D2596d28
    0.0    1.0    0.0   16.0  0.0  0.0    0.0   47.4   0   5 c4t50060482D52D2596d96
    0.0   10.0    0.0  103.9  0.0  1.2    0.0  123.0   0  94 c5
    0.0    1.0    0.0    8.0  0.0  0.0    0.0   41.2   0   4 c5t50060482D52D2599d9
    0.0    4.0    0.0   40.0  0.0  0.5    0.0  123.3   0  49 c5t50060482D52D2599d6
    0.0    1.0    0.0    8.0  0.0  0.0    0.0    2.5   0   0 c5t50060482D52D2599d3
    0.0    3.0    0.0   32.0  0.0  0.6    0.0  215.1   0  36 c5t50060482D52D2599d28
    0.0    1.0    0.0   16.0  0.0  0.0    0.0   47.5   0   5 c5t50060482D52D2599d97
```

Ack!

```
Iostat -xnCz 1 | grep c
extended device statistics
    r/s    w/s   kr/s   kw/s wait actv wsvc_t asvc_t  %w  %b device
    6.0    8.0   48.0   88.0 13.0  0.1  928.6    6.4  50   4 c2
   19.0    8.0  152.0   88.0  0.0  0.1    0.0    4.5   0   5 c3
   93.0    5.0 1616.0   17.5  0.0  1.1    0.0   10.8   0  11 c6
   83.0    5.0  696.0   33.0  0.0  0.8    0.0    9.0   0   8 c7
    0.0    4.0    0.0   56.0 13.0  0.0 3250.0    9.6 100   3 c2t0d0
    6.0    4.0   48.0   32.0  0.0  0.1    0.0    5.0   0   4 c2t1d0
    0.0    4.0    0.0   56.0  0.0  0.0    0.0   11.1   0   4 c3t0d0
   19.0    4.0  152.0   32.0  0.0  0.1    0.0    3.4   0   7 c3t1d0
   16.0    0.0  128.0    0.0  0.0  0.2    0.0   13.4   0  21 c6t24d2
   19.0    0.0  152.0    0.0  0.0  0.2    0.0   11.6   0  22 c6t24d3
```

iostat oddness

- Iostat seems to be a bit funny on Solaris 10 sometimes
  - I have seen controllers' stats for %b go well over 100
  - This appears to be a bug

iostat on a busy machine

```
extended device statistics
    r/s    w/s   kr/s   kw/s wait actv wsvc_t asvc_t  %w  %b device
    1.0  181.5    7.9  563.3  0.1  2.8    0.5   15.3   1  82 c0
    1.0  181.5    7.9  563.3  0.1  2.8    0.5   15.3   3  82 c0t0d0
   82.4    0.0 27226.5    0.0  0.0  2.5    0.0   29.9   0 197 c1
    2.9    0.0 1012.7    0.0  0.0  0.1    0.0   31.4   0   6 c1t50060482CC369DD8d217
    0.0    0.0    0.0    0.0  0.0  0.0    0.0    0.0   0   4 c1t50060482CC369DD8d204
    9.8    0.0 4922.4    0.0  0.0  0.4    0.0   43.3   0  39 c1t50060482CC369DD8d185
   22.6    0.0 3776.2    0.0  0.0  0.3    0.0   14.0   0  23 c1t50060482CC369DD8d177
   15.7    0.0 3211.0    0.0  0.0  0.3    0.0   21.2   0  26 c1t50060482CC369DD8d120
   16.7    0.0 7788.1    0.0  0.0  0.8    0.1   50.6   0  63 c1t50060482CC369DD8d119
    2.9    0.0 1012.8    0.0  0.0  0.1    0.0   31.9   0   7 c1t50060482CC369DD8d115
    5.9    0.0 2504.4    0.0  0.0  0.1    0.0   21.5   0  11 c1t50060482CC369DD8d110
    2.0    0.0  997.1    0.0  0.0  0.1    0.0   38.9   0   8 c1t50060482CC369DD8d88
  264.0    5.9 50616.6    5.9  0.0  6.4    0.0   23.8   0 289 c2
    2.0    0.0  510.3    0.0  0.0  0.0    0.0   23.2   0   5 c2t50060482CC369DD7d217
    2.9    0.0 1004.9    0.0  0.0  0.1    0.0   35.5   0   7 c2t50060482CC369DD7d194
   27.5    0.0 4365.2    0.0  0.0  0.5    0.0   17.2   0  33 c2t50060482CC369DD7d185
    1.0    0.0  502.5    0.0  0.0  0.0    0.0   33.3   0   3 c2t50060482CC369DD7d183
   26.5    0.0 5040.4    0.0  0.0  0.5    0.0   17.1   0  32 c2t50060482CC369DD7d177
    7.9    0.0 2959.9    0.0  0.0  0.3    0.0   32.7   0  22 c2t50060482CC369DD7d120
   58.9    0.0 9052.4    0.0  0.0  2.1    0.0   35.7   0  61 c2t50060482CC369DD7d119
    4.9    0.0 1523.1    0.0  0.0  0.2    0.0   34.1   0  12 c2t50060482CC369DD7d115
   23.6    0.0 1232.6    0.0  0.0  0.4    0.0   16.3   0   8 c2t50060482CC369DD7d110
    0.0    2.0    0.0    2.0  0.0  0.0    0.0    0.8   0   0 c2t50060482CC369DD7d105
    6.9    0.0 1837.2    0.0  0.0  0.2    0.0   24.5   0  12 c2t50060482CC369DD7d88
```

sar

- Sar –d output example. Note that the output has been passed through grep –v to omit lines with all zero statistics.

```
> sar -d 1 5 | grep -v "0     0.0       0       0     0.0     0.0"
SunOS king 5.10 Generic_118833-33 sun4u    02/18/2008
11:14:40    device        %busy   avque   r+w/s  blks/s  avwait  avserv
            md8               0     0.0       1      16     0.0     2.2
            md9               0     0.0       1      16     0.0     3.6
            md104             0     0.0       1      16     4.6     3.7
            sd15              1     0.0       2      17     0.0     4.0
            sd15,d            0     0.0       1       1     0.0     4.5
            sd15,f            0     0.0       1      16     0.0     3.6
            ssd0              1     0.0       2      17     0.0     2.8
            ssd0,d            0     0.0       1       1     0.0     3.5
            ssd0,f            0     0.0       1      16     0.0     2.1
            ssd11             7     0.2       3      32     0.0    70.5
            ssd11,c           7     0.2       3      32     0.0    70.5
            ssd15             0     0.0       1       1     0.0     3.9
            ssd15,c           0     0.0       1       1     0.0     3.9
            ssd54             7     0.1       1       1     0.0    69.9
            ssd54,c           7     0.1       1       1     0.0    69.9
            ssd58            14     0.1       1      31     0.0   143.8
            ssd58,c          14     0.1       1      31     0.0   143.8
            ssd64             0     0.0       1      31     0.0     2.3
            ssd64,c           0     0.0       1      31     0.0     2.3
            ssd67             9     0.1       1      31     0.0    93.8
            ssd67,c           9     0.1       1      31     0.0    93.8
```

Network

- Batch things with arraysize to reduce trips
- Know your requirements
  - Does it have to run on the app server?
- Know your thresholds
  - Acceptable latency times [Millsap (2003)]

SQL*Net transmission via WAN	200 ms (.2 seconds)
SQL*Net transmission via LAN	15 ms (.015 seconds)
SQL*Net transmission via IPC	1 ms (.001 seconds)

Network tools

- netstat

```
watson:/u01/home/oracle> netstat -irn 3
    input   eri0      output           input  (Total)    output
packets errs  packets errs  colls  packets errs  packets errs  colls
792058224 0     860444922 6238  0      259604881 204267 -310421021 6238  0
1378    0     1578    0     0      4301    0     4483    0     0
1028    0     1172    0     0      5800    0     5925    0     0
527     0     546     0     0      6419    0     6413    0     0
480     0     506     0     0      6827    0     6834    0     0
490     0     543     0     0      5098    0     5132    0     0
```

Note that this is packet data

Does not show how big the packets are

No utilization data

Network tools, cont'd

- nicstat
  - Part of the K9Toolkit
  - Available from bredangregg.com/K9Toolkit/nicstat.c

```
Int             Interface
rKb/s           read Kbytes/s
wKb/s           write Kbytes/s
rPk/s           read Packets/s
wPk/s           write Packets/s
rAvs            read Average size, bytes
wAvs            write Average size, bytes
%Util           %Utilisation (r+w/ifspeed)

Sat             Saturation (defer, nocanput, norecvbuf, noxmtbuf)
```

Nicstat, cont'd

```
> gcc -o nicstat nicstat.c -l kstat -l rt -l gen

> nicstat -z 1
    Time   Int   rKb/s   wKb/s   rPk/s   wPk/s    rAvs    wAvs   %Util      Sat
09:39:44   ce0   95.62 4855.74 1550.10 3292.90   63.17 1510.00    4.06     0.00
09:39:44  eri0    2.77    0.94    3.60    3.50  789.39  275.59    0.03     0.00
    Time   Int   rKb/s   wKb/s   rPk/s   wPk/s    rAvs    wAvs   %Util      Sat
09:39:45   ce0    0.67    0.00    5.00    0.00  136.80    0.00    0.00     0.00
09:39:45  eri0 1043.51  345.61 1331.00 1283.00  802.82  275.84   11.38     0.00
    Time   Int   rKb/s   wKb/s   rPk/s   wPk/s    rAvs    wAvs   %Util      Sat
09:39:46   ce0    0.35    0.00    5.00    0.00   71.20    0.00    0.00     0.00
09:39:46  eri0 1050.25  348.00 1358.00 1313.00  791.94  271.40   11.45     0.00
```

truss

- Used to trace a process' system calls
- Handy switches

```
-a      shows arg strings from each exec() call
-d      shows time stamp on each line, relative to beginning
        of the trace, in seconds.fraction format
-D      show time delta
-f      follow all children (fork() or vfork()) of the truss'd
        process
-o      output file the trace is written to
-p      process ID to be traced
```

- Example syntax:

```
truss -adDf -o <outfile> -p <pid>
truss -adDf -o /tmp/truss_orcl.out -p m12345
```

truss example

```
 time fsof  elapsed    syscall & parms
23606:     10.7471      0.0001     fcntl(13, F_DUPFD, 0x00000100)                                  = 257
23606:     10.7473      0.0002     close(13)                                                       = 0
23606:     10.7474      0.0001     fcntl(257, F_SETFD, 0x00000001)                                 = 0
23606:     10.7475      0.0001     fstatvfs(257, 0xFFFFFFFF7FFF6208)                               = 0
23606:     10.8782      0.1307     pread(257, "0602\0\006C0 B FC8 <07 $".., 524288, 0x10918000) = 524288
23606:     11.1139      0.2357     pread(257, "0602\0\006C0 B fF0BEBF\n".., 507904, 0x10998000) = 507904
23606:     11.2242      0.1103     pread(256, "0602\0\00681B2C6EB91A1 @".., 524288, 0x6CB18000) = 524288
23606:     11.3268      0.1026     pread(256, "0602\0\00681B2E6F0BEBF0E".., 507904, 0x6CB98000) = 507904
23606:     11.4325      0.1057     pread(257, "0602\0\006C0 B86DCDE +FF".., 524288, 0x10A18000) = 524288
23606:     11.5063      0.0738     pread(257, "0602\0\006C0 BA6F0BEBF H".., 507904, 0x10A98000) = 507904
23606:     11.5702      0.0639     pread(256, "0602\0\00681B306F0BEBF L".., 524288, 0x6CC18000) = 524288
23606:     11.6475      0.0773     pread(256, "0602\0\00681B3 &F0BEBF O".., 507904, 0x6CC98000) = 507904
23606:     11.7339      0.0864     pread(257, "0602\0\006C0 BC6EE DE4 F".., 524288, 0x10B18000) = 524288
23606:     11.8250      0.0911     pread(257, "0602\0\006C0 BE6EB85 FD1".., 507904, 0x10B98000) = 507904
23606:     11.8646      0.0396     pread(256, "0602\0\00681B3 FF0BEBF W".., 524288, 0x6CD18000) = 524288
23606:     11.9016      0.0370     pread(256, "0602\0\00681B3 fF0BFC9\b".., 507904, 0x6CD98000) = 507904
23606:     12.0232      0.1216     pread(257, "0602\0\006C0 C06EE DE6F0".., 524288, 0x10C18000) = 524288
23606:     12.2208      0.1976     pread(257, "0602\0\006C0 C &F0BFC914".., 507904, 0x10C98000) = 507904
23606:     12.3829      0.1621     pread(256, "0602\0\00681B386F0BEBF `".., 524288, 0x6CE18000) = 524288
23606:     12.5267      0.1438     pread(256, "0602\0\00681B3A6EB85 FF9".., 507904, 0x6CE98000) = 507904
23606:     12.6472      0.1205     pread(257, "0602\0\006C0 C FF0BFC9  ".., 524288, 0x10D18000) = 524288
23606:     12.6921      0.0449     pread(257, "0602\0\006C0 C fF0BEBF g".., 507904, 0x10D98000) = 507904
```

References & additional info

McDougall, R.; Mauro, J.; Gregg, B. 2006. *Solaris Performance and Tools: DTrace and MDB techniques for Solaris 10 and OpenSolaris*, Prentice Hall

Millsap, C. V.; Holt, J. L. 2003. *Optimizing Oracle Performance*, O'Reilly: Sebastopol CA

Various Unix man pages (truss, vmstat, iostat, etc)

Items Learned in this Session

- We have looked at brief overviews for some of included and freely available tools for investigating Solaris performance and capacity metrics
  - You should have learned:
    - Available tools for Solaris performance investigation
    - Some indicators of capacity issues recognizable with these tools
  - You should be able to investigate Solaris performance issues more efficiently, and get an idea as to if you are nearing capacity

Questions?

Thank You!

- Thank you for attending this presentation
- Please provide feedback via your evaluation form
- Session information:
  - Dallas Deeds, Solaris Performance Diagnostics for Database People
- Contact information for further questions
  - deedsd@nationwide.com
    - Please put "NYOUG Solaris Performance" in the subject line